UNLOCKING BANDWIDTH FOR GPUS IN CC-NUMA SYSTEMS

Neha Agarwal*
David Nellans
Mike O’Connor
Stephen W. Keckler
Thomas F. Wenisch*

NVIDIA
University of Michigan*

(Major part of this work was done when Neha Agarwal was an intern at NVIDIA)

HPCA-2015
EVOLVING GPU MEMORY SYSTEM

GDDR5  200 GB/s  GPU  150.8 GB/s  CPU  80 GB/s  DDR4

Roadmap

CUDA 1.0-5.x
cudaMemcpy
Programmer controlled copying to GPU memory

CUDA 6.0+: Current
Unified virtual memory
Run-time controlled copying  Better productivity

Future
CPU-GPU cache-coherent high BW interconnect

How to best exploit full BW while maintaining programmability?

HPCA-2015
DESIGN GOALS & OPPORTUNITIES

- Simple programming model:
  - No need for explicit data copying

- Exploit full DDR + GDDR BW
  - Additional 30% BW via NVLink
  - Crucial to BW sensitive GPU apps

Design intelligent dynamic page migration policies to achieve both these goals

HPCA-2015
BANDWIDTH UTILIZATION

- Coherence-based accesses, no page migration
- Wastes GPU memory BW

<table>
<thead>
<tr>
<th>Bandwidth (GB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GDDR5</td>
</tr>
<tr>
<td>200</td>
</tr>
<tr>
<td>DDR4</td>
</tr>
<tr>
<td>80</td>
</tr>
</tbody>
</table>

- NVLink 80 GB/s
- GPU Memory BW 200 GB/s
- Total System Memory BW 280 GB/s
- BW from CC 80 GB/s

HPCA-2015
BANDWIDTH UTILIZATION

Static Oracle: Place data in the ratio of memory bandwidths [ASPLOS’15]

Dynamic migration can exploit the full system memory BW
Excessive migration leads to under-utilization of DDR BW

Migrate pages for optimal BW utilization
CONTRIBUTIONS

Intelligent Dynamic Page Migration

- Aggressively migrate pages upon First-Touch to GDDR memory
- Pre-fetch neighbors of touched pages to reduce TLB shootdowns
- Throttle page migrations when nearing peak BW

Dynamic page migration performs 1.95x better than no migration
Comes within 28% of the static oracle performance
6% better than Legacy CUDA
OUTLINE

- Page Migration Techniques
  - First-Touch page migration
  - Range-Expansion to save TLB shootdowns
  - BW balancing to stop excessive migrations

- Results & Conclusions
FIRST-TOUCH PAGE MIGRATION

- Naive: Migrate pages that are touched

First-Touch migration approaches Legacy CUDA

First-Touch migration is cheap, no hardware counters required
PROBLEMS WITH FIRST-TOUCH MIGRATION

- TLB shootdowns may negate benefits of page migration

How to migrate pages without incurring shootdown cost?

Throughput Relative To No Migration

<table>
<thead>
<tr>
<th></th>
<th>backprop</th>
<th>needle</th>
<th>pathfinder</th>
</tr>
</thead>
<tbody>
<tr>
<td>No overhead</td>
<td>1.5</td>
<td>2.0</td>
<td>1.5</td>
</tr>
<tr>
<td>TLB shootdown</td>
<td>1.0</td>
<td>1.5</td>
<td>1.0</td>
</tr>
</tbody>
</table>

HPCA-2015
PRE-FETCH TO AVOID TLB SHOOTDOWNS

- Intuition: Hot virtual addresses are clustered
- Pre-fetch pages before access by the GPU

No shootdown cost for pre-fetched pages
MIGRATION USING RANGE-EXPANSION

- Pre-fetch pages in spatially contiguous range

<table>
<thead>
<tr>
<th></th>
<th>CPU memory</th>
<th>GPU memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accessed, shootdown required</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Pre-fetched, shootdown not required</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Throughput Relative to No Migration:

- No range expansion

```
<table>
<thead>
<tr>
<th></th>
<th>No overhead</th>
<th>TLB shootdown</th>
<th>No overhead</th>
<th>TLB shootdown</th>
<th>No overhead</th>
<th>TLB shootdown</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>backprop</td>
<td>needle</td>
<td>pathfinder</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

HPCA-2015
MIGRATION USING RANGE-EXPANSION

- Pre-fetch pages in spatially contiguous range

Range-Expansion hides TLB shootdown overhead
REVISITING BANDWIDTH UTILIZATION

First-Touch + Range-Expansion aggressively unlocks GDDR BW

How to avoid excessive page migrations?

HPCA-2015
BANDWIDTH BALANCING

Throttle migrations when nearing peak BW

HPCA-2015
BANDWIDTH BALANCING

Throttle migrations when nearing peak BW

% of Migrated Pages to GPU Memory vs. Total Bandwidth (GB/s)

- Excessive migrations
- First-Touch migrations

Target

- DDR under-subscribed
- GDDR under-subscribed

% of Accesses from GPU Memory vs. Time

First-Touch + Range Exp vs. First-Touch + Range Exp + BW Balancing

Start

End

HPCA-2015
SIMULATION ENVIRONMENT

- Simulator: GPGPU-Sim 3.x
- Heterogeneous 2-level memory
  - GDDR5 (200GB/s, 8-channels)
  - DDR4 (80GB/s, 4-channels)
- GPU-CPU interconnect
  - Latency: 100 GPU core cycles
- Workloads:
  - Rodinia applications [Che’IISWC2009]
  - DoE mini apps [Villa’SC2014]
RESULTS

Throughput Relative to No Migration

- Legacy CUDA
- First-Touch + Range Exp + BW Balancing
- Static Oracle

HPCA-2015
FIRST-TOUCH + RANGE-EXPANSION + BW BALANCING OUTPERFORMS LEGACY CUDA

HPCA-2015
RESULTS

Streaming accesses, no-reuse after migration

HPCA-2015
RESULTS

Dynamic page migration performs 1.95x better than no migration
Comes within 28% of the static oracle performance
6% better than Legacy CUDA

HPCA-2015
CONCLUSIONS

- Developed migration policies without any programmer involvement
- First-Touch migration is cheap but has high TLB shootdowns
- First-Touch + Range-Expansion technique unlocks GDDR memory BW
- BW balancing maximizes BW utilization, throttles excessive migrations

These 3 complementary techniques effectively unlock full system BW
THANK YOU
EFFECTIVENESS OF RANGE-EXPANSION

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Execution Overhead of TLB Shootdown</th>
<th>% Migrations Without Shootdown</th>
<th>Execution Runtime Saved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backprop</td>
<td>29.1%</td>
<td>26%</td>
<td>7.6%</td>
</tr>
<tr>
<td>Pathfinder</td>
<td>25.9%</td>
<td>10%</td>
<td>2.6%</td>
</tr>
<tr>
<td>Needle</td>
<td>24.9%</td>
<td>55%</td>
<td>2.75%</td>
</tr>
<tr>
<td>Mummer</td>
<td>21.15%</td>
<td>13%</td>
<td>2.75%</td>
</tr>
<tr>
<td>Bfs</td>
<td>6.7%</td>
<td>12%</td>
<td>0.8%</td>
</tr>
</tbody>
</table>

Range-Expansion can save up to 45% TLB shootdowns
DATA TRANSFER RATIO

Performance is low when GDDR/DDR ratio is away from optimal
GPU WORKLOADS: BW SENSITIVITY

GPU workloads are highly BW sensitive

HPCA-2015
Dynamic page migration performs 1.95x better than no migration. Comes within 28% of the static oracle performance. 6% better than Legacy CUDA.

HPCA-2015